Listing of the Claims:

1. (original) A method for implementing a hardware controller that concurrently executes a number of tasks by carrying out operations on behalf of the tasks, the method comprising:

determining a format for a context, comprising stored information related to a task, that represents the task;

determining possible states, and transitions between states, that a context representing a task currently executed by the hardware controller can occupy at each point in the execution of the task, transitions representing operations performed on behalf of a task by the hardware controller;

partitioning the states and operations carried out by the hardware controller into a number of managers each containing a number of related states and carrying out a number of operations;

associating each manager with a data structure for storing contexts occupying states contained by the manager;

defining a data-structure-manipulator manager that implements the data structures and that transfers contexts from one data structure to another;

defining a command interface to the data-structure-manipulator manager for each manager; and

implementing the managers and data-structure-manipulator manager, according to the determined states and transitions, so that, when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator manager to transfer the context from the data

structure associated with the first manager to the data structure associated with the second manager.

2. (original) The method of claim 1 wherein tasks are provided to the hardware controller via a signal interface, wherein the hardware controller generates output signals and output data as a result of execution of a task, and wherein operations carried out by managers can be invoked via a signal interface provided for each manager.

3. (previously amended) The method of claim1 wherein the data-structure-manipulator manager comprises a manipulator logic circuit for each manager, a manipulator logic circuit for a manager together with the command interface defined for the manager composing a manipulator within the data-structure-manipulator manager corresponding to the manager.

4. (previously amended) The method of claim 3 wherein the hardware controller includes a timing circuit that generates clock signals ~~are~~ at regular intervals, the intervals including and following a first clock signal and preceding a next clock signal composing a clock cycle, wherein each manager can issue a single command to the manipulator associated with the manager during a single clock cycle, and wherein each manipulator can receive a number of contexts during a single clock cycle for transfer to the data structure associated with the manager corresponding to the manipulator.

5. (original) The method of claim 4 wherein each manipulator can receive a number of commands from a number of managers during each clock cycle.

6. (original) The method of claim 5 wherein related contexts can be linked to one another to form a chain of related contexts that can be transferred together by transferring the first context of the chain of related contexts.

7. (original) The method of claim 6 applied to an outbound sequence manager functionality of a fibre channel interface controller to implement an outbound sequence manager having contexts that store information supplied to the outbound sequence manager in outbound descriptor blocks and having doubly linked-list data structures associated with each manager for storing contexts occupying states contained in the manager, the outbound sequence manager comprising:

a completion manager associated with a completion doubly linked-list;

a credit manager associated with a timer doubly linked-list;

a transmit manager associated with a transmit doubly linked-list;

an outbound descriptor block manager associated with a free doubly linked-list;

a rogue manager associated with a free doubly linked-list;

a non-fabric daemon manager associated with a non-fabric doubly linked-list; and

a centralized list manager data-structure-manipulator manager that transfers contexts from one doubly linked-list to another in response to

commands issued to the centralized list manager by the completion manager, credit manager, transmit manager, outbound descriptor manager, rogue manager, and non-fabric daemon, the centralized list manager having a timer list manipulator, a free list manipulator, a non-fabric list manipulator, a transmit list manipulator, and a completion list manipulator.

8. (original) A method for implementing a hardware controller that concurrently executes a number of tasks, the method comprising:

representing each task executed by the hardware controller as a context, each context occupying a state determined by the contents of at least one field within the context, a context transitioning from one state to another state when the hardware controller carries out an operation on behalf of the task represented by the context;

partitioning hardware controller operations and associated context states into a number of logical managers;

associating each logical manager with one of a number logical data structures for storing contexts occupying states within the logical manager; and

implementing the logical managers and a data-structure manipulator that contains the contexts, logical data structures, and a command interface through which each logical manager issues commands to direct the data-structure manipulator to transfer a context from the data structure associated with the logical manager to a different data structure.

9. (original) The method of claim 8 wherein timing of the hardware controller is controlled by a clock circuit that generates clock signals that define

clock cycles, wherein each logical manager may issue at most one command to the data-structure manipulator during each clock cycle, and wherein the data-structure manipulator concurrently executes commands issued by the logical managers during a clock cycle by serializing the commands according to a defined precedence ordering of the commands.

10. (original) The method of claim 8 wherein the data structures are chosen for efficient storage and retrieval of contexts according to the operations carried out by one or more logical managers associated with the contexts, the data structures chosen from among well-known data structures employed in software programming, including:

singly linked lists;

doubly linked lists;

first-in-first-out queues;

first-in-last-out queues;

stacks;

Graphs;

acyclic graphs, such as binary trees;

arrays;

circular queues; and

combinations of the well-known data structures.

11. (original) The method of claim 8 wherein each logical manager is associated with a signal interface for input and output of signals, wherein operations carried out by logical managers are invoked by signals received

through the signal interfaces, and wherein the hardware controller receives tasks and control signals and output data and control signals though a hardware controller interface.

12. (original) The method of claim 11 wherein the hardware controller receives a task via the hardware controller interface and executes the task by:

storing information related to the task within the hardware controller and initializing a context to represent the task;

adding the context to the data structure;

carrying out operations on behalf of the context by the logical manager associated with the data structure in which the context is located, and, when carrying out an operation by a first logical manager results in transition of the context to a state in a second logical manager associated with a different data structure than the data structure in which the context is located, issuing a command from the first logical manager to the data-structure manipulator to transfer the context to the different data structure; and

when all operations that need to be carried out by the hardware controller to execute the task are carried out, generating output data and output signals corresponding to completion of the task by the hardware controller and freeing the context for representing a subsequently received task.

13. (previously amended) A subcomponent controller within a communication controller comprising:

data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller;

logical managers that are each associated with a data structure and that each carries out operations on behalf of contexts stored within the associated data structure; and

a data-structure manipulator that implements a number of data structures for storing contexts and that transfer contexts between data structures in response to receiving transfer commands from the logical managers.

14. (original) The subcomponent controller of claim 13 wherein the subcomponent interfaces with external subcomponent controllers via a signal interface and wherein the subcomponent controller receives timing signals at regular intervals that define clock cycles.

15. (original) The subcomponent controller of claim 14 wherein each logical manager may issue a single context transfer command during a single clock cycle, wherein the data-structure manipulator can concurrently receive and carry out one transfer command received from each logical manager during a single clock cycle, wherein the data-structure manipulator serializes all commands received during a single clock cycle by carrying out the commands logically in a predetermined precedence order.

16. (original) The subcomponent controller of claim 13 wherein the communications controller is a fibre channel interface controller and wherein the communications medium is a fibre channel communications medium.

17. (original) The subcomponent controller of claim 16 wherein the subcomponent controller is an outbound sequence manager that receives outbound descriptor blocks from an external subcomponent, stores information related to an outbound sequence and represents a received outbound sequence with a context, and that provides fibre channel frames to an external subcomponent for transmission to the fibre channel medium as a result of executing a task corresponding to a received outbound descriptor block.

18. (original) The subcomponent controller of claim 17 wherein the data structures are doubly linked lists of contexts in which each context may reference a single linked list of related contexts.

19. (previously amended) The subcomponent controller of claim 18 wherein the logical managers include:

a completion manager associated with a completion list;

a credit manager associated with a timer list;

a transmit manager associated with a transmit list;

an outbound descriptor block manager associated with a free list;

a rogue manager associated with a free list;

a non-fabric daemon manager associated with a non-fabric list; and

a centralized list manager that serves as the data-structure manipulator to transfer contexts between lists.

20. (original) The subcomponent controller of claim 19 wherein:

the completion manager can issue commands to the centralized list manager to transfer a context from the completion list to the free list, to transfer a first context from the completion list to the free list and dechain related contexts from the first context and transfer the related contexts to the transmit list, to transfer a first context from the completion list to the free list and dechain related contexts from the first context and transfer related contexts to the completion list, and to transfer a first context from the completion list to the free list and to dechain related contexts from the first context and transfer related contexts to the non-fabric list;

the credit manager can issue commands to the centralized list manager to transfer a context from the timer list to the transmit list, non-fabric list, or the completion list;

the transmit manager can issue commands to the centralized list manager to transfer a context from the transmit list to the timer list or the completion list;

the outbound descriptor block manager can issue commands to the centralized list manager to transfer a context from the free list to the transmit list of the non-fabric list; and

the non-fabric daemon can issue commands to the centralized list manager to transfer a context from the non-fabric list to the transmit list.